УДК 004.031.42

А. Л. ЕРОХИН, канд. техн. наук,
А. В. КОЛЬЧЕНКО

МЕТОДИКА СБОРА ПОЛЬЗОВАТЕЛЬСКОЙ ИНФОРМАЦИИ ДЛЯ ОЦЕНКИ ЮЗАБИЛИТИ КОМПЬЮТЕРНЫХ ИНТЕРФЕЙСОВ

Оцінка юзабіліті інтерфейсів комп'ютерних систем проводиться на основі обробки даних користувача. Розглянуто особливості взаємодії користувачів з web-додатками з точки зору подієвого підходу. Запропонована методика збору користувальницької інформації при роботі з web-додатками. Розроблене агентно-орієнтоване клієнт-серверне програмне забезпечення, що реалізує розглянуту методику.

Usability evaluation of the computer interfaces is based on the user data processing. The features of interaction between user and web-based applications from the point of event approach are considered. The method for collection user data interface events is offered. Agent-oriented client-server software realizing the considered method is developed.

Введение. Актуальной проблемой является проектирование и построение интеллектуальных пользовательских web-интерфейсов человеко-машинных систем. Международный стандарт ISO9421 определяет понятие «юзабилити» и предъявляет набор требований к интерфейсам нового поколения [1]. Соответствие интерфейсов стандарту устанавливается в результате тестирования. В процессе тестирования необходимо производить сбор и анализ информации, поступающей от пользователя системы. Получаемая таким образом информация также используется при построении и модификации моделей взаимодействия пользователя с web-приложениями. Таким образом, задача сбора пользовательской информации является важной и требует разрешения.

Постановка задачи. События, возникающие при взаимодействии пользователя с web-приложениями, в дальнейшем будем называть пользовательскими событиями. В работе [2] приводится обзор и классификация как пользовательских событий, так и подходов к их обработке. В настоящей работе ограничимся рассмотрением пользовательских событий, генерируемых устройствами ввода (клавиатура, мышь) и событиями элементов управления оконной системы (получение/потеря фокуса ввода, прокрутка содержимого окна и т.д.). Эти события возникают в процессе взаимодействия пользователя с программным обеспечением-клиентом (браузером) и контентом web-приложения и отражают поведенческую активность пользователя.

Сбор пользовательской информации в специальных лабораториях обходится достаточно дорого. Другой подход предполагает удаленный сбор

пользовательской информации [3]. Его преимуществами являются: вовлечение в процесс тестирования практически неограниченного количества пользователей, непрерывный характер и естественные условия проведения.

Существующий инструментарий для сбора пользовательской информации обладает рядом недостатков. Так, система [4] является устаревшей, поскольку не учитывает новые стандарты объектной модели DOM, а также имеет недостатки в части хранения собираемой информации. Инструментарий [5] не является достаточно гибким и имеет ограничения по способу организации транзакций с сервером. Другие средства сбора пользовательской информации не являются универсальными либо требуют больших затрат для их применения.

Целью настоящей работы является разработка методики и реализация инструментария для извлечения пользовательской информации при работе с web-приложениями.

Требования к методике. С точки зрения событийного (бихевиористического) подхода сеанс пользователя с web-приложением выглядит как множество событий, связанных с активностью пользователя. Вначале происходит загрузка необходимого приложения с локального или удаленного компьютера. Для решения своих задач, находясь в среде web-приложения, пользователь воздействует на его элементы. К этим воздействиям относятся перемещения мыши, передача и потеря фокуса ввода элементами управления, нажатия клавиш на клавиатуре, прокрутка контента web-приложения, щелчки мыши на объектах приложения – элементах управления, изображениях, гиперссылках. Набор этих действий, их последовательность при решении специфических задач образуют *поведенческие паттерны* пользователя. Любое из действий пользователя и соответствующее ему событие может оказаться важным для распознавания существующего или построения нового паттерна. Таким образом, одним из важных требований является возможность регистрировать все возможные пользовательские события, отображающие активность пользователя при взаимодействии с web-приложением.

В то же время не во всех случаях необходим сбор всей пользовательской информации. Для проведения различных исследований, включая оценку юзабилити, необходимо собирать ту или иную информацию. Этот факт определяет требование гибкости конфигурации инструментария для сбора пользовательской информации.

Разработчики web-приложений должны иметь возможность простого и удобного использования инструментария для сбора пользовательской информации в новых приложениях. Кроме того, необходимо обеспечить возможность встраивания такого инструментария в уже существующие приложения, при этом важным является критерий минимизации затрат на их модификацию.

Web-приложения характеризуются возможностью выполнения на различных программно-аппаратных платформах – от традиционных персональных компьютеров до мобильных устройств, поэтому необходимо обеспечить возможность работы создаваемого инструментария независимо от платформы.

Собираемую пользовательскую информацию необходимо сохранять. Таким образом, возникают задачи спецификации формата файла данных и способа организации транзакций с сервером при выполнении сохранения. Формат файла должен предусматривать хранение необходимой и неизбыточной информации о сеансе работы пользователя. Транзакции с сервером должны выполнятся в асинхронном режиме, чтобы не прерывать текущий сеанс пользователя и экономить сетевой трафик.

Кроме того, инструментарий для сбора пользовательской информации должен иметь хотя бы минимальный набор функций для обработки пользовательской информации. Эти функции обеспечивают фильтрацию пользовательских событий, очистку их от «шума» и первичный анализ «сырых» данных.

Итак, сформулируем основные требования к инструментарию для сбора пользовательских событий:

1) регистрация различных событий, поступающих от пользователя;
2) гибкость и простота конфигурирования;
3) возможность встраивания в существующие приложения;
4) кроссплатформенность;
5) удобный формат представления регистрируемых событий;
6) набор функций обработки и анализа пользовательской информации.

Методика сбора пользовательских данных. Средой выполнения большинства web-приложений являются браузеры. Web-приложения реализуются на языках HTML и XML и представляют собой web-документы. Все современные браузеры являются DOM-конформными. DOM представляет собой независимый от платформы и языка интерфейс, который обеспечивает программам и скриптам динамический доступ и изменение содержания, структуры и стиля документов. DOM предоставляет стандартный набор объектов, представляющих документы HTML и XML, стандартную модель взаимодействия этих объектов, а также стандартный интерфейс для доступа к этим объектам и методы для манипуляции с ними.

Спецификация DOM Events определяет общую независимую от платформы и языка систему событий, которая позволяет регистрировать обработчики событий, описывает прохождение события по древовидной структуре документа и предоставляет базовую контекстуальную информацию для каждого события.

Таким образом, спецификацию DOM целесообразно выбрать в качестве основы методики сбора пользовательских данных. Сама же методика заключается в выполнении следующей последовательности действий.

1. Определение ключевых элементов web-приложения, от которых необходимо собирать пользовательские данные. В web-документе этими элементами являются конкретные элементы управления, изображения, области документа, оформленные в виде соответствующих тегов на языке HTML. Окно браузера также может выступать в качестве такого элемента.
2. Определение для каждого ключевого элемента набора событий, мониторинг которых необходимо осуществлять – получение или потеря фокуса ввода, нажатия клавиш на клавиатуре, перемещения мыши, нажатия кнопок мыши и т.д.
3. Определение для каждого ключевого элемента контекстуальной информации, которую необходимо извлекать при срабатывании пользовательского события. Так, при выборе гиперссылки часто бывает необходимо извлечь адрес запрашиваемого ресурса.
4. Установка фильтров на собираемые пользовательские данные для фильтрации «шумов» и предотвращения избыточности собираемых данных. Фильтры могут ограничивать как временные интервалы мониторинга, так и пространственные. Так, при мониторинге перемещений мыши в среде web-приложения необходимо определять минимальный пройденный указателем путь. Тогда объем собираемой информации уменьшается в несколько раз без потери семантики.
5. Встраивание в web-приложение программного агента для упаковки собираемых данных в соответствии с форматом файла данных и передачи их в асинхронном режиме, «прозрачно» для пользователя на сервер. Этот же агент может производить и ряд других интеллектуальных функций обработки «сырых» пользовательских данных. Наличие такого агента позволяет не поддерживать постоянную связь с сервером, что существенно повышает быстродействие системы и экономит сетевой трафик.

Выполнение указанных действий обеспечивает внедренного программного агента своеобразным «зрением» (рис. 1). Но в отличие от пользователя, воспринимающего весь контент web-приложения, агент «видит» только ключевые элементы приложения и выполняет мониторинг и обработку активности пользователя. Возможность конфигурирования системы позволяет создавать различные модификации агентов, соответствующие различным поведенческим паттернам с web-приложением.

Область применения методики. Регистрируемые в процессе взаимодействия пользователя и приложения события сохраняются для дальнейшей обработки. Учитывая то, что пользовательские данные имеют сложную структуру, а также возможность обработки собранных данных как автоматизированном, так и в «ручном» режимах, возникает необходимость в

определении стандарта формата файла, хранящего собранные в течении сессии данные пользователя.

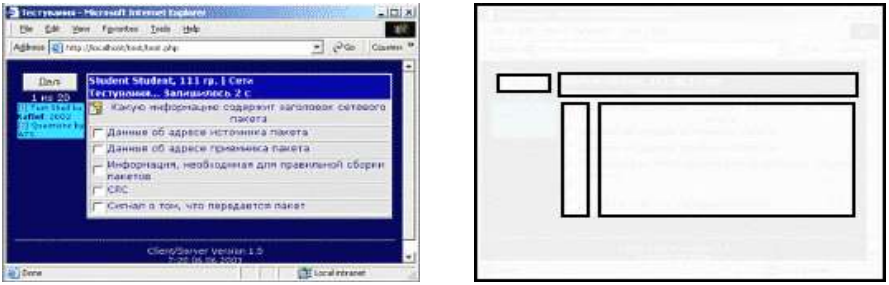

Рис. 1. Восприятие web-приложения пользователем (слева) и агентом (справа)

Каждое регистрируемое событие сопровождается пространственно-временными характеристиками – меткой времени и координатами в пределах окна web-приложения.

Отметим, что для каждого пользовательского события, кроме факта его возникновения, необходимо регистрировать и сохранять дополнительную контекстуальную информацию. При последующей обработке такая информация поможет спроецировать низкоуровневые пользовательские события (движения мыши, нажатия кнопок) на уровень семантики выполняемой пользователем задачи (поиск информации, тестирование знаний). Проекция уровней решения задачи показана в таблице.

Таблица

Проекция уровней решения задачи

Уровни решения задачи	**Характеристика**
1. Семантический (проблемно-ориентированный)	Описание задачи в терминах предметной области («пройти тестирование знаний», «выбрать правильный ответ»)
2. Логические события	Описание в терминах интерфейсных элементов приложения («нажать кнопку ДАЛЕЕ», «отметить радиокнопку»)
3. Низкоуровневые физические события	Описание в терминах моторики пользователя («переместить указатель мыши от текущего местоположения на кнопку, опустить левую кнопку мыши вниз, поднять левую кнопку мыши вверх»)

Предложенная методика позволяет регистрировать события, возникающие на уровнях логических событий и низкоуровневых физических событий. Эти же события доступны для мониторинга программному агенту, внедренному в web-приложения.

Реализация инструментария для сбора пользовательской информации. Разработанная методика реализована в виде программного продукта. Для выделения ключевых элементов web-приложения и определения для них набора событий для мониторинга, а также для указания сопровождаемой контекстуальной информации разработан специальный язык разметки. Его языковые конструкции вставляются разработчиком как атрибуты тегов HTML, которые представляют ключевые элементы web-приложения. Разбор объектов и прохождение по дереву DOM приложения выполняет внедренный скрипт, реализованный на языке javascript. Фильтры для низкоуровневых событий от устройств ввода – клавиатуры и мыши, также реализованы в виде скрипта. Программный агент для сбора и обработки пользовательских данных, а также для выполнения транзакций с сервером реализован в виде flash-объекта. Выбор средства его реализации связан с поддержкой объектов flash на многих программно-аппаратных платформах. Взаимодействие с сервером производится с помощью объекта XML flash-ролика, что обеспечивает асинхронный, независимый от сеанса пользователя с web-приложением режим передачи информации с сервером. На клиентской стороне объект flash взаимодействует с web-приложением с помощью прикладного программного интерфейса, специфицированного фирмой Macromedia.

Выводы. Предложенная методика реализует сбор информации для построения поведенческого паттерна человека-оператора. Не до конца решенной остается задача спецификации формата файла событий, хранящего пользовательскую информацию, собранную во время сеанса взаимодействия пользователя с web-приложением. Следующей задачей является построение на основе обработки пользовательской информации паттернов взаимодействия пользователя с web-приложением.

Список литературы: **1**. *Travis D.* Bluffers' Guide to ISO 9241. **2**. *Hilbert D.M.*. A Survey of Computer-Aided Techniques for Extracting Usability Information from User Interface Events. **3**. *Hartson H.R., Castillo J.C., Kelso J.* Remote Evaluation: The Network as an Extension of the Usability Laboratory // CHI 96. Conference on Human Factors in Computing Systems, 1996. **4**. *Etgen M., Cantor J.* What does getting WET (Web Event-logging Tool) Mean for Web Usability? // 5th Conference on Human Factors & the Web, 1999. **5.** *Ерохин А.Л., Кольченко А.В.* О создании систем тестирования знаний с адаптацией на испытуемого // Образование и виртуальность. – Выпуск 7. – Харьков-Ялта, УАДО, ХНУРЭ, 2003. – С. 299-304.

Поступила в редакцию 14.04.04